

UNE APPROCHE PAR RECHERCHE ARBORESCENTE POUR AMÉLIORER DES POLITIQUES ISSUES D’UN APPRENTISSAGE PAR RENFORCEMENT

Laurent Péret, Frédérick Garcia

Institut National de la Recherche Agronomique - Unité Biométrie et Intelligence
Artificielle
{peret , fgarcia}@toulouse.inra.fr

Résumé

Les algorithmes standard de l'apprentissage par renforcement sont traditionnellement centrés autour de la notion de fonction de valeur. Une alternative à l'apprentissage d'une fonction de valeur consiste à effectuer une recherche directe dans l'espace des politiques. Nous avons exploré une autre voie, encore peu exploitée en apprentissage par renforcement, qui, sur la base d'un arbre de décision, optimise "on-line" les politiques déduites d'une fonction de valeur, celle-ci étant éventuellement de médiocre qualité. Ces différentes méthodes sont illustrées sur un problème complexe de gestion d'une constellation de satellites. Notre solution est basée sur une modélisation du problème de maintenance et de déploiement comme un problème décisionnel de Markov. Les résultats que nous présentons dans cet article concernent l'optimisation de la phase de maintenance, les phases de déploiement et de fin de service étant supposées déterminées. Deux méthodes distinctes d'optimisation ont été mises en œuvre dans un premier temps : (i) l'itération de la politique approchée où la politique est évaluée par simulation, (ii) l'optimisation stochastique, où une politique experte paramétrée est directement optimisée par simulation. Nous évaluons par la suite une technique par recherche arborescente ayant pour objectif d'améliorer on-line la première approche.

Mots-clés : Apprentissage par renforcement, optimisation stochastique, planification, incertitude, constellation de satellites.

1 INTRODUCTION

L'apprentissage par renforcement (Sutton & Barto, 1998) permet à un agent autonome d'adapter son comportement à un environnement incertain, en procédant par essais et erreurs. Ce processus vise à produire une politique optimale, c'est-à-dire l'action la mieux adaptée à chaque situation envisageable pour le problème

décisionnel considéré. Classiquement, l'apprentissage par renforcement est basé sur une *fonction de valeur*. Divers algorithmes et structures de représentation ont été proposés pour apprendre cette fonction de valeur dans le cadre formel des *processus décisionnels markoviens* (PDMs). Les PDMs sont un modèle probabiliste du monde définissant espaces d'états et d'actions, la dynamique du système et les récompenses accumulées par l'agent au cours du temps. Ils se sont imposés ces dernières années en intelligence artificielle comme un standard pour la résolution de problèmes de décision séquentielle sous incertitude. Les méthodes d'apprentissage basées sur la fonction de valeur ou méthodes *purement critique*¹ exploitent efficacement cette description markovienne du problème pour approcher la politique optimale. Néanmoins, une représentation compacte du problème est requise pour assurer la convergence de ces méthodes pour des PDMs de grande taille en un temps raisonnable. Ceci suppose que l'on puisse exhiber une structure particulière pour le PDM considéré. En l'absence d'une telle structure, les méthodes classiques de l'apprentissage sont alors souvent vouées à l'échec.

Récemment ont émergé des méthodes d'apprentissage (Baxter & Bartlett, 1999; Marbach & Tsitsiklis, 1998) ne faisant pas appel à la fonction de valeur pour apprendre une politique optimale-approchée. Ces méthodes, et leurs bonnes performances observées dans nombre de cas, remettent en cause le caractère essentiel de la fonction de valeur dans l'apprentissage par renforcement. Afin de mieux comprendre le rôle joué par la présence - ou non - de la fonction de valeur au sein des méthodes d'apprentissage, nous menons des travaux autour d'une application de très grande taille, centrée sur la maintenance d'une constellation de satellites. Dans ce cadre, différentes expériences ont été réalisées sur plusieurs instances de ce problème :

(i) une approche *purement critique* consistant à apprendre sur l'espace d'états une fonction de valeur approximative V_ω par *itération de la politique approchée*. Une politique π est déduite en évaluant pour tout état la fonction de valeur "à un coup", c'est-à-dire en développant les états successeurs immédiats de l'état considéré pour chaque action possible sur un horizon d'une étape de décision.

(ii) une seconde approche consistant à apprendre une politique optimale sans chercher à approximer la fonction de valeur optimale : on effectue la recherche sur la politique elle-même. Ces méthodes *purement acteur* consistent à optimiser la performance d'une politique paramétrée π_θ , l'espace de recherche est alors l'espace des paramètres Θ . Ces techniques s'affranchissent de la tâche difficile qu'est le maintien d'une fonction de valeur pour se concentrer sur l'amélioration de la performance de la politique.

A partir de premiers résultats (Garcia *et al.*, 2001), nous avons montré que (i) ne peut lutter efficacement avec (ii) pour le problème considéré. La conclusion principale était que pour des problèmes de très grande taille, en l'absence d'une structure particulière, une approximation moyenne de la fonction de valeur ne

1. On distingue dans l'architecture décisionnelle des systèmes d'apprentissage par renforcement la partie "agissante" ou l'acteur, chargé de déterminer à chaque instant l'action à effectuer, de la partie "critique" représentée par la fonction de valeur et chargée d'évaluer les choix de l'acteur au regard de l'expérience accumulée

permettait pas de décider efficacement et que les approches de type (ii) devaient être préconisées. Le présent article prolonge cette étude et remet partiellement en cause ces premières conclusions, la mise en œuvre de techniques d'optimisation sur (i) permettant d'approcher et même de dépasser les résultats obtenus par (ii). L'amélioration de (i) considérée développe les états sur un *horizon temporel* de longueur supérieure à une seule étape de décision (Davies *et al.*, 1998). Cette recherche nécessite de simuler les trajectoires associées aux différentes séquences d'actions, autrement dit de développer un *arbre de décision* dont la profondeur est la longueur de l'horizon temporel. La fonction de valeur peut alors servir à valuer les feuilles de l'arbre ainsi généré. Le prix à payer pour cette amélioration est un surcoût en calcul, exponentiel en la taille de l'horizon temporel.

Le plan de l'article est le suivant : après avoir rappelé le cadre des PDMs nous présentons les principes des deux grandes familles d'algorithmes d'apprentissage, ceux purement critique, basés sur la fonction de valeur et ceux purement acteur, basés sur une politique paramétrée, puis nous introduisons le principe de recherche arborescente permettant d'améliorer l'approche par fonction de valeur. Le problème de la maintenance d'une constellation de satellites est alors présenté, et les différents algorithmes employés précisément décrits. La dernière section analyse les résultats numériques obtenus et tire quelques conclusions.

2 PROCESSUS DÉCISIONNELS DE MARKOV

Les *PDMs* (*Processus Décisionnels de Markov* (Puterman, 1994)) permettent de modéliser la dynamique d'un agent en interaction avec un environnement stochastique. A chaque *instant* t , l'agent perçoit l'*état* courant de l'environnement s_t , et détermine son *action* courante a_t conformément à une *politique* π . L'agent reçoit alors une *récompense* r_t , fonction de l'état et de l'action courants. L'environnement évolue alors vers l'état suivant s_{t+1} , selon l'effet conjugué de sa dynamique propre et de l'action a_t . Un PDM est défini par un quintuplet $\langle S, A, T, P, R \rangle$ où :

- (i) S est l'ensemble de tous les *états* possibles du systèmes;
- (ii) A est l'ensemble de toutes les *actions* pouvant être effectuées;
- (iii) T est l'ensemble ordonné des *instants* auxquels les décisions peuvent être prises, c'est-à-dire l' *horizon temporel global*;
- (iv) P définit les *probabilités de transition* entre toute paire d'états appartenant à S après l'exécution d'une action appartenant à A ;
- (v) R définit les *coûts* ou *revenus instantanés* associés à ces transitions.

On complète la description du problème par la propriété de Markov, vérifiée par la dynamique de l'environnement : s_{t+1} ne dépend que de s_t et de a_t , autrement dit le processus est sans mémoire. Par ailleurs, on suppose généralement la stationnarité de l'environnement, c'est-à-dire que les probabilités de transition n'évoluent pas avec le temps t .

2.1 Politiques et critères

Une *politique* est définie comme une fonction π qui associe à tout état $s \in S$ et à tout instant $t \in T$ une *action* $a_t = \pi(s, t)$. Dans ce cadre, l'objectif poursuivi est de trouver une *politique optimale*, c'est-à-dire une fonction π^* définissant pour tout état $s \in S$ et pour tout instant $t \in T$ une *action optimale* $a_t^* = \pi^*(s, t)$, minimisant l'espérance du *coût global* (ou maximisant l'espérance du *revenu global*) sur l'horizon temporel restant. Le coût ou revenu global peut être défini comme la *somme γ -pondérée* (où γ est le facteur d'actualisation, pouvant être interprété comme un taux d'inflation) des coûts ou revenus instantanés $r_t = r(s_t, a_t)$ associés aux transitions effectuées : $J_\pi = \sum_{t \in T} \gamma^t r_t$. Si une politique ne dépend pas de l'instant t mais seulement de l'état s , elle est alors dite *stationnaire*.

2.2 Fonction de valeur et optimalité

La théorie des PDMs associe à toute *politique* π une *fonction de valeur* V_π qui associe à chaque état $s \in S$ et à chaque instant $t \in T$ l'espérance du coût ou du revenu global $V_\pi(s, t)$, obtenue en suivant π depuis s et t . Par exemple, la fonction de valeur associée au critère γ -pondéré s'écrit pour une politique stationnaire π :

$$V_\pi(s) = E_\pi \left[\sum_{t \in T} \gamma^t r_t \mid s_0 = s \right] \quad (1)$$

où E_π désigne l'espérance mathématique lorsque la politique π est appliquée.

Les *équations d'optimalité de Bellman* (équation 2 (Bellman, 1957)) caractérisent de manière compacte l'*unique fonction de valeur optimale* V^* de laquelle une politique optimale est déduite (équation 3). Dans le cas d'un coût global défini sur un horizon infini, ces équations sont les suivantes :

$$V^*(s) = \min_{a \in A} [r(s, a) + \gamma \cdot \sum_{s' \in S} P(s' | s, a) \cdot V^*(s')] \quad (2)$$

$$\pi^*(s) = \operatorname{argmin}_{a \in A} [r(s, a) + \gamma \cdot \sum_{s' \in S} P(s' | s, a) \cdot V^*(s')] \quad (3)$$

C'est de la notion d'optimalité des PDMs que découle la fonction de valeur V qui permet de définir simplement la politique optimale.

3 SIMULATION ET OPTIMISATION POUR LES PDMs

Pour les PDMs avec des ensembles S et A finis et de petite taille, l'*itération de la valeur* ou l'*itération de la politique* sont des algorithmes de la *programmation dynamique* qui exploitent les données et la structure du problème avec efficacité

pour calculer cette fonction de valeur optimale et en déduire une politique optimale associée. Mais, les méthodes de la programmation dynamique ne sont applicables qu'à des problèmes de dimension relativement modeste d'une part et dont le *modèle* - autrement dit les probabilités de transitions P - est entièrement connu d'autre part. En effet, ces algorithmes nécessitent de pouvoir stocker en mémoire S , A et P et leurs temps de résolution sont polynomiaux en les tailles de S et A .

L'apprentissage par renforcement permet de contourner ces difficultés et de résoudre des problèmes de grande taille issus du monde réel par la mise en œuvre de deux principes :

- la *représentation paramétrique* des fonctions de valeur et des politiques par des approximations compactes permettant d'aborder des problèmes décisionnels de grande taille ;
- l'emploi de la *simulation* pour estimer les grandeurs inconnues et éviter des calculs inabordables en un temps raisonnable.

Aujourd'hui, l'apprentissage par renforcement est l'une des principales approches utilisées pour résoudre des problèmes de décision séquentielle avec des transitions de probabilités inconnues et/ou des PDMs de grande taille. On peut classer les différents algorithmes d'apprentissage dans deux familles de méthodes, présentées ci-dessous.

3.1 Apprentissage purement critique

Les algorithmes "historiques" de l'apprentissage par renforcement comme le *Q-learning*, ou *Sarsa* (Sutton & Barto, 1998), sont exclusivement basés sur la fonction de valeur, qui définit la politique suivie (cf. équation 4). Ces algorithmes peuvent être interprétés comme des versions stochastiques et simulées des méthodes de la programmation dynamique. Pour les problèmes de grande taille, excluant de pouvoir stocker en mémoire les réels $V(s)$ pour tous les états du système, on utilise habituellement une représentation paramétrique du type $V_w(s)$ où w est un vecteur de paramètres de taille raisonnable - par exemple, les poids d'un réseau de neurones. On recherche une approximation de la fonction de valeur optimale de laquelle est déduite une politique proche de la politique optimale.

Ainsi, ont été proposés différents algorithmes pour apprendre V approximée par $V_w(s)$. Si le modèle sous-jacent est supposé connu, une politique π est dérivée de $V_w(s)$ suivant l'équation 4 :

$$\pi(s) = \operatorname{argmin}_{a \in A} [r(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) \cdot V_w(s')] \quad (4)$$

Si le modèle n'est pas connu, on apprend plutôt (dans le cas du Q-learning par exemple), une fonction de valeur $Q_w(s,a)$ associée aux *paires état-action* de laquelle une politique π est dérivée suivant l'équation 5 :

$$\pi(s) = \operatorname{argmin}_{a \in A} [Q_w(s, a)] \quad (5)$$

Ainsi, l'apprentissage d'une fonction de valeur V_w suffit à déterminer rapidement l'action à exécuter.

3.2 Apprentissage purement acteur

Des travaux récents (Anderson, 1999) ont toutefois montré que la fonction de valeur n'est pas nécessairement l'outil le mieux adapté pour apprendre un comportement optimal-approché : il est parfois plus efficace d'effectuer des recherches *directement* sur la politique suivie par l'agent, sans chercher à valuer chaque état du système par une fonction de valeur. On considère dans ce cadre des *politiques paramétrées* par un vecteur $\theta \in \Theta \subset R^p$. Une politique paramétrée π_θ peut être structurée suivant un ensemble de règles ou par des distributions de probabilité sur l'espace des actions. Les algorithmes de l'apprentissage purement acteur optimisent la performance $E[J]$ d'une politique, définie comme l'espérance du coût global d'une trajectoire, en particulier par des approches de type gradient. Le cœur de ces méthodes consiste alors à estimer par simulation le gradient de la performance par rapport à θ .

Ces méthodes d'apprentissage peuvent être décrites dans le cadre de l'*optimisation stochastique* : on cherche le minimum d'une fonction à valeurs réelles stochastique J d'un vecteur θ de p paramètres réels. Ainsi, si $\Theta \subseteq R^p$ est le domaine du vecteur paramètre θ , le problème est de trouver une valeur $\theta^* \in \Theta$ qui minimise l'espérance de J :

$$\theta^* = \operatorname{argmin}_{\Theta} E[J(\theta)] \quad (6)$$

Ces méthodes furent originellement développées dans le cadre des réseaux de neurones connectionnistes dans les années 80 et au début des années 90 (Williams, 1992). Elles ont connu un vif regain d'intérêt depuis quelques années avec notamment les travaux de Baxter et Bartlett (Baxter & Bartlett, 1999). La plupart des méthodes de recherche directe proposées (Baxter & Bartlett, 1999; Marbach & Tsitsiklis, 1998) sont basées sur l'estimation du gradient de $E[J]$ par simulation. Elles exploitent la structure markovienne du problème et s'inscrivent dans la logique d'apprentissage par essais et erreurs : typiquement, chaque action déterminée par la politique courante π_θ apporte une "petite" contribution pour estimer le gradient de J par rapport à θ . Cette contribution traduit l'impact bénéfique ou néfaste de l'action considérée au regard des récompenses ou coûts subséquents.

Dans certains cas, il peut être préférable de se placer à un niveau de granularité supérieur, en particulier sur des espaces d'états de grande taille, continus ou partiellement observables (Strens & Moore, 2001). Dans ce cas, la structure markovienne du problème est abstraite et le simulateur est considéré comme une

boîte noire associant à une valeur de θ une observation bruitée de J . Diverses méthodologies (Azavadar, 1999) ont été proposées pour résoudre ce problème. Ces techniques d'optimisation sont très diverses, selon la structure et la taille de l'espace des paramètres Θ : cette classe de méthodes regroupe des algorithmes de type évolutionniste adaptés à des espaces discrets et d'autres spécifiques à des espaces continus.

Dans le cas d'une méthode de type gradient, l'évolution du paramètre θ est régie selon une règle de la forme :

$$\theta_{n+1} = \theta_n - \alpha_n \hat{\nabla} J(\theta_n) \quad (7)$$

où α_n est le pas de l'algorithme et $\hat{\nabla} J(\theta_n)$ un estimateur du gradient de J en θ_n obtenu par simulation. L'une des forces des approches purement acteur est la garantie d'une convergence vers un optimum local de la fonction J .

Entre la famille de méthodes "purement critique" présentée à la section précédente et celle des méthodes "purement acteur" présentée ci-dessus ont été récemment proposés des algorithmes "acteur-critique" (Sutton *et al.*, 1999) qui trouvent leurs racines dans des travaux plus anciens (Barto *et al.*, 1983). Ces algorithmes optimisent une politique paramétrée tout en maintenant une fonction de valeur associée à chaque état. La fonction de valeur n'intervient plus alors pour déterminer le choix d'une action mais sert à accélérer la convergence des algorithmes d'optimisation en diminuant la variance des estimateurs du gradient. Le bénéfice apporté par la fonction de valeur peut s'interpréter en termes d'accumulation de connaissances sur le système ; les approches "purement acteur" n'autorisent pas une telle mémorisation : toutes les données obtenues par simulation sont "oubliées" entre deux valeurs consécutives de θ .

3.3 Recherche arborescente on-line

On distingue traditionnellement en apprentissage par renforcement le temps de simulation ou temps "off-line", durant lequel une politique est apprise au cours de trajectoires simulées, du temps d'exécution "on-line", en général bref, durant lequel la politique apprise est appliquée sans être remise en cause. Une alternative peu exploitée dans le cadre de l'apprentissage par renforcement consiste à consentir à un effort de calcul important en cours d'exécution pour améliorer la politique apprise. L'idée maîtresse présentée dans cette section repose sur l'exploration on-line depuis l'état courant s_t d'ensembles de trajectoires simulées. On abandonne l'ambition de calculer off-line une politique optimale-approchée pour tous les états du système qui peut se révéler être une tâche d'une difficulté insurmontable. La principale conséquence pratique est que le calcul préalable à la prise de décision on-line peut être très long.

La politique classiquement suivie par l'agent pour une fonction de valeur donnée V consiste à évaluer pour l'état courant s_t toutes les actions possibles sur un pas de temps - seuls les états successeurs immédiats de l'état courant sont

envisagés (cf. équations 4 et 4). Une extension naturelle pour les problèmes de décision séquentielle consiste à développer *un arbre de décision* en se projetant dans le futur sur un horizon H . On examine alors les conséquences des séquences d'actions $\{a_t, a_{t+1}, \dots, a_{t+H-1}\}$ en simulant les trajectoires associées $\{s_t \rightarrow s_{t+1} \rightarrow \dots \rightarrow s_{t+H}\}$. Dans un cadre déterministe plus proche de la planification classique, où le but de l'agent est d'atteindre un *état-cible*, Davies et al (Davies *et al.*, 1998) mettent en pratique ce principe. Ils intègrent dans leurs algorithmes une fonction de valeur approximative V qui value les feuilles de l'arbre développé - dans le cas où $V = V^*$, $H = 1$ suffit à déterminer l'action optimale. Leurs résultats expérimentaux sur des problèmes de petite dimension font apparaître l'avantage décisif procuré par cette recherche arborescente.

Dans le cas stochastique, Kearns et al (Kearns *et al.*, 1999) proposent un algorithme procédant par échantillonnage, en ne considérant dans un état donné pour une action donnée que C trajectoires dont les récompenses sont moyennées. Contrairement à Davies et al, Kearns et al. ne présupposent pas l'existence d'une fonction de valeur. Une fonction de valeur déduite du parcours complet de l'arbre est alors redéfinie pour l'état courant s de manière récursive par :

$$V_H(s) = \begin{cases} 0 & \text{si } H = 1 \\ \min_{a \in A} [r(s, a) + \gamma \frac{1}{C} \sum_{s' \in S} V_{H-1}(s')] & \text{sinon} \end{cases}$$

où l'on génère par simulation C états s' pour une action a et un état s donnés.

Le résultat théorique établi par Kearns et al relie la complexité de l'algorithme et l'erreur commise sur la fonction de valeur déduite du parcours complet de l'arbre de décision. Il établit que l'on peut rendre $\|V_H - V^*\|_\infty = \max_{s \in S} |V_H(s) - V^*(s)|$ arbitrairement petit en choisissant C et H suffisamment grands, sans autre hypothèse que l'existence d'un simulateur markovien du système (V^* étant la fonction de valeur optimale). La complexité par évaluation de V_H est alors en $O(|A|C^H)$.

Nous avons cherché à appliquer à notre problème de très grande taille ces principes de recherche arborescente on-line en vue notamment d'améliorer les politiques obtenues par apprentissage purement critique.

4 APPLICATION À LA MAINTENANCE OPTIMALE D'UNE CONSTELLATION DE SATELLITES

Les constellations de satellites (SKYBRIDGE, GALILEO . . .) sont des programmes spatiaux complexes aux coûts exorbitants qui posent de nouveaux problèmes en termes de conception et de planification. En raison de l'emploi de technologies de pointe et de l'hostilité du milieu spatial, les pannes survenant au cours des phases de déploiement ou d'exploitation sont relativement fréquentes. Dans un contexte de forte concurrence commerciale, les retards dans l'ouverture du service et les périodes d'indisponibilité partielle sont très coûteuses.

L'objectif poursuivi est de produire des stratégies pour le déploiement et la maintenance qui soient optimales vis-à-vis d'un critère de coût tout en tenant compte des aléas susceptibles de survenir. Différents types de pannes peuvent survenir pendant le déploiement ou la phase opérationnelle : échec lors d'un lancement, panne d'un satellite, fin de vie d'un satellite... En raison des délais importants imposés par la préparation et le lancement d'un satellite, des satellites de rechange ("spares") sont disponibles sur des orbites de réserve pour pallier d'éventuelles pannes de satellites opérationnels. Sur ces orbites de réserve, les spares dérivent par rapport aux orbites opérationnelles, à raison d'une orbite par mois. Comme les satellites opérationnels, ces satellites sont sujets aux pannes.

4.1 Un Problème Décisionnel de Markov

Le déploiement et la maintenance d'une constellation de satellites peuvent être envisagés comme un problème de décision séquentielle dans l'incertain, modélisable comme un PDM complètement observable (Garcia *et al.*, 2001):

- les décisions doivent être prises séquentiellement pour la préparation et le lancement de nouveaux satellites, et pour la mise à poste en orbite opérationnelle des spares ;
- les conséquences d'une décision ne sont pas connues avec certitude, en raison de l'éventuelle occurrence de pannes ;
- on peut néanmoins supposer que l'état du système à un instant donné est connu avec certitude.

Ainsi :

(i) S est défini par un ensemble de *variables d'état* et leur *domaines* respectifs ; les variables d'états retenues sont le *nombre* de satellites opérationnels et de spares sur chaque plan orbital, et l'*âge* de chaque satellite. Pour tenir compte des décisions à effet différé, il faut également intégrer dans l'état le *plan* de préparation et de lancement des satellites défini sur un *horizon de décision* et décrit par le *nombre* des satellites en préparation, la *date de lancement* et le *type de lanceur* pour les tirs planifiés sur l'horizon de décision ;

(ii) de même, A est défini par un ensemble de *variables de décision* et leurs *domaines* respectifs ; les variables de décision *immédiatement exécutables* sont le *plan orbital* en cas de lancement, et le *nombre des spares* mis à poste en orbite opérationnelle ; les variables de décision à *effet différé* concerne la *décision de lancement* définie par le *nombre de satellites* à mettre sur orbite ; de plus, les contraintes entre état et variables de décision nous permettent de spécifier quelles décisions peuvent être appliquées pour un état donné ;

(iii) T est *discrétisé* et est considéré comme *fini*; le pas de temps entre deux instants successifs du processus décisionnel est égal à un mois (durée mise par un spare pour dériver d'un plan orbital au suivant) ;

(iv) P est défini par un ensemble de *variables aléatoires* associées aux pannes possibles et par les *probabilités conditionnelles* entre variables d'état à un instant

donné, les variables de décision associées au même instant, les variables aléatoires associées à la transition consécutive à la décision, et les variables d'état associées à l'instant suivant;

(v) R définit les *coûts instantanés* associés aux *états* (indisponibilité partielle ou totale de la constellation) et aux *décisions* (coûts de préparation et de lancement);

(vi) le *coût global* est défini comme la somme γ -pondérée des coûts instantanés, et l'on recherche une politique *non stationnaire* qui minimise l'espérance du coût global. Cela implique que la fonction de valeur de l'apprentissage par renforcement dépend de l'instant de l'horizon temporel considéré. Plus précisément, la fonction de valeur est définie par une famille de fonctions de valeur $V_t, t \in T$. L'adaptation du cas stationnaire présenté dans la section 3 ne pose pas de difficultés majeures pour les différents algorithmes d'apprentissage.

Nous avons dégagé deux caractéristiques spécifiques de ce PDM, essentielles pour aborder sa résolution :

- présence d'un plan courant entraînant la distinction entre deux classes de variables d'état et de décision
- non stationnarité : au cours de la maintenance, la constellation traverse une phase de renouvellement nécessitant de nombreux tirs sur une courte période de temps. Lorsque tous les satellites initialement mis sur orbite arrivent en fin de vie, il est alors nécessaire de redéployer la quasi-intégralité des satellites.

En raison du grand nombre de décisions possibles à chaque instant, nous introduisons un ordre arbitraire entre les décisions : d'abord préparation de nouveaux satellites et extension du plan de lancement, ensuite choix du plan orbital en cas de lancement, et enfin, mise à poste des spares en orbite opérationnelle. Ainsi, chaque instant est artificiellement décomposé en trois étapes et nous introduisons 3 fonctions de valeur, 3 équations d'optimalité simplifiées et 3 sous-politiques. Cette décomposition du problème est illustrée par la figure 1. Malgré cette simplification de l'espace des actions, le problème est encore très complexe en raison de l'immense taille des espaces d'états.

4.2 Cas de référence

Les résultats présentés dans cet article concernent l'optimisation de la phase de maintenance, les phases de déploiement et de fin de service étant supposées définies.

Nous considérons une constellation de 32 satellites opérationnels, régulièrement répartis entre 8 plans orbitaux. Trois types de lanceurs sont disponibles avec une capacité d'emport de 2, 4 et 6 satellites. Nous supposons que les probabilités d'échec au lancement et de panne satellite sont connues avec certitude : la probabilité mensuelle de panne est supposée constante et égale à 0,03 ; les probabilités d'échec lanceur sont supposées être égales à respectivement 0.10, 0.08 et 0.05 pour les lanceurs à 2, 4 et 6 satellites. La durée de vie d'un satellite est

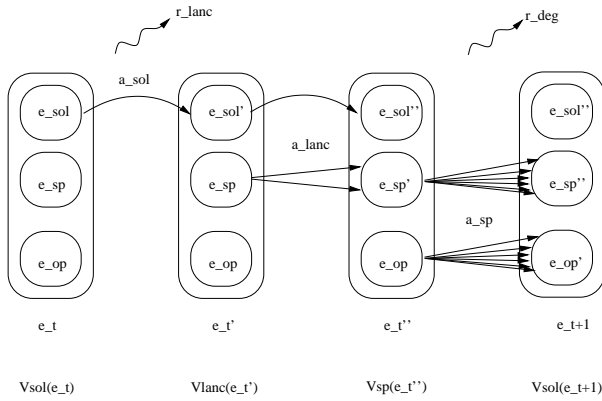


FIG. 1 – Décomposition séquentielle du problème de décision en 3 étapes

supposée être de 6 ans. Un satellite coûte 1 unité ; les lanceurs à 2, 4, et 6 satellites coûtent respectivement 1, 1.5 et 2 unités. Les coûts de dégradation sont de 1 unité par satellite opérationnel en panne et par mois. Dans l'état initial de la phase de maintenance, tous les satellites sont supposés être opérationnels et âgés de 0 mois. Nous simulons des trajectoires sur un horizon de 10 ans. Le facteur d'actualisation mensuel γ vaut 0.99.

La taille de l'espace d'états est de 10^{123} pour le problème de référence. Si l'on peut évaluer les résultats fournis par les techniques de résolution d'apprentissage par renforcement décrites aux sections 3.1 et 3.2, les temps d'exécution requis par la recherche arborescente ne permettent pas de valider cette dernière approche. Pour pouvoir démontrer son intérêt, nous avons défini une instance réduite de notre problème, qui reste toutefois de très grande taille par rapport aux PDMs habituellement considérés en pratique. Les caractéristiques des deux problèmes sont résumées dans le tableau 1.

	N_S	N_P	Panne	d_{ans}	T_{ans}	$ S $
Cas complet	32	8	0.003	6	10	10^{123}
Cas réduit	4	2	0.03	4	5	10^{17}

TAB. 1 – Caractéristiques des deux instances du problème de maintenance

4.3 Apprentissage purement critique

Nous avons retenu l'algorithme de l'itération de la politique approchée (Bertsekas & Tsitsiklis, 1996), qui peut être utilisé lorsque les transitions de probabilités du PDM sont connues, comme c'est le cas pour notre application.

Cet algorithme consiste en l'alternance de deux phases successives qui sont répétées itérativement. A chaque itération k de l'algorithme, (i) la fonction de

valeur V_{π_k} , associée à la politique courante π_k est estimée, et (ii) une nouvelle politique π_{k+1} est dérivée de V_{π_k} en utilisant l'équation 8.

$$\pi_{k+1}(s) = \operatorname{argmin}_{a \in A} [r(s,a) + \gamma \cdot \sum_{s' \in S} P(s'|s,a) \cdot V_{\pi_k}(s')] \quad (8)$$

La fonction de valeur, associée à une politique π est définie par l'équation 9.

$$V_{\pi}(s) = E[\sum_{t \in T} \gamma^t r(s_t, \pi(s_t)) | s_0 = s] \quad (9)$$

Mais, pour les PDMs de grande taille, ce calcul ne peut pas être mené de manière exacte. La fonction de valeur V_{π} est approximée par une fonction V_w . La première phase de chaque itération de l'algorithme consiste à estimer les paramètres w_k tels que V_{w_k} approxime au mieux V_{π_k} . Ceci est fait par échantillonnage, en simulant n trajectoires de longueur l , en partant de divers états initiaux et en suivant la politique π_k . Si s_{0_i} est l'état initial de la i -ème trajectoire, w_k est défini par l'équation 11.

$$w_k = \operatorname{argmin}_w \sum_{i=1}^n [V_w(s_{0_i}) - \sum_{t=0}^{l-1} \gamma^t \cdot r(s_t, \pi_k(s_t)) | s_0 = s_{0_i}]^2 \quad (10)$$

L'algorithme $TD(\lambda)$ (Bertsekas & Tsitsiklis, 1996) a été proposé pour calculer le vecteur w_k en mettant à jour les poids incrémentalement après chaque transition. La suite de vecteurs w_k ainsi définie converge vers \hat{w} , tel que la politique associée $\hat{\pi}$ est une politique approchant la politique optimale.

La qualité du résultat dépend fortement de la qualité de l'approximation V_w . Une architecture fréquemment mise en œuvre est une simple *approximation linéaire*: $V_w(s) = \sum_{j=1}^m w^j \cdot \phi_j(s)$, où w^j est la j -ième composante de w et où $\phi_1(s), \dots, \phi_m(s)$ sont m fonctions réelles caractéristiques de l'état du système. Le choix de ces fonctions caractéristiques est crucial pour la performance de l'algorithme: elles doivent résumer le plus efficacement possible l'état du système.

La première phase de la première itération de l'algorithme consiste à estimer la fonction de valeur associée à la politique experte de départ π_0 (voir section 4.3), en simulant un ensemble de trajectoires obtenues en suivant π_0 et en minimisant l'erreur quadratique entre les coûts observés et la fonction de valeur paramétrée (voir équation 10).

Pour notre application, nous sommes incapables de stocker les trois fonctions de valeur définies dans la section 4.1 dans un tableau qui aurait pour dimension la taille de l'espace d'états. C'est pourquoi nous définissons une paramétrisation de l'état, par une approximation linéaire de chaque fonction de valeur basée sur une

centaine de fonctions caractéristiques de l'état (une vingtaine pour le problème réduit) : des caractéristiques élémentaires telles le nombre et les âges des satellites opérationnels et des spares pour chaque plan orbital, et des caractéristiques synthétiques, telles l'âge moyen des satellites pour chaque plan orbital. Nous avons mis en œuvre l'algorithme $TD(\lambda)$ dont la convergence vers un minimum (local) de l'erreur quadratique est obtenue après quelques itérations.

4.4 Apprentissage purement acteur

Nous avons mis en œuvre une procédure classique d'optimisation stochastique basée sur le gradient, la méthode de Kiefer-Wolfowitz due à (Robbins & Monro, 1951) et (Kiefer & Wolfowitz, 1952), qui utilise des observations bruitées de la fonction J . Son intérêt réside dans le peu d'hypothèses requises sur le système, considéré comme une boîte noire, et dans les garanties de convergence vers un minimum local qu'elle possède. Une version projetée de cet algorithme itératif est définie par l'équation 11:

$$\theta_{n+1} = \pi(\theta_n + \alpha_n \frac{1}{2\beta_n} (J(\theta_n + \beta_n) - J(\theta_n - \beta_n))) \quad (11)$$

où, pour tout $\theta \in R^p$, $\pi(\theta)$ est le point de Θ le plus proche de θ , et où α_n et β_n sont des réels positifs satisfaisant aux conditions suivantes :

$$\sum_1^\infty \alpha_n = \infty, \quad \lim_{n \rightarrow \infty} \beta_n = 0, \quad \sum_1^\infty \alpha_n \beta_n < \infty, \quad \sum_1^\infty \frac{\alpha_n^2}{\beta_n^2} < \infty$$

A chaque itération de l'algorithme, la fonction J est estimée par simulation. L'algorithme est stoppé lorsqu'un certain critère d'arrêt est satisfait (par exemple un nombre maximal d'itérations est atteint).

La politique que nous tentons d'optimiser est basée sur un ensemble de *règles expertes* paramétrées. Ces règles ont été mises au point suite à des échanges avec des experts du domaine spatial, capables d'inférer une décision raisonnable en fonction de quelques paramètres du système. Une règle paramétrée est définie pour chacune des trois étapes de décision du processus décisionnel (voir figure 1). Ces règles sont paramétrées via 7 paramètres réels. Par exemple, la règle pour le remplacement d'un satellite opérationnel implique un paramètre qui est l'âge maximum au-delà duquel un satellite doit être remplacé si un spare est disponible sur le même plan orbital.

4.5 Recherche arborescente

L'algorithme mis en œuvre pour la recherche arborescente utilise la meilleure fonction de valeur V obtenue par itération de la politique pour valuer les feuilles de l'arbre de décision. D'autre part, il intègre la connaissance du modèle dont

nous disposons. La fonction de valeur permettant de dériver une politique s'écrit alors :

$$V_H(s) = \begin{cases} V(s) & \text{si } H = 1 \\ \min_{a \in A} [r(s,a) + \gamma \cdot \sum_{s' \in S} P(s'|s,a) \cdot V_{H-1}(s')] & \text{sinon} \end{cases} \quad (12)$$

La première étape de décision ne comportant pas d'aléa, chaque action d'extension du plan de lancement ne conduit qu'à un unique état. La deuxième étape comporte un aléa qui est l'échec éventuel du lanceur : chaque action de mise en orbite peut donc mener à deux états différents. Pour ces deux premières étapes de décision, le nombre d'actions possibles étant faible, on peut donc calculer récursivement la fonction de valeur exactement comme indiqué dans l'équation 13. Pour la dernière étape de décision, intégrant les pannes satellites, le nombre d'états accessibles est variable mais en général élevé. Afin de pouvoir développer l'arbre de décision sur une profondeur de plusieurs mois, nous avons choisi de supprimer l'aléa concernant les pannes satellites². Nous nous basons ainsi sur une hypothèse déterministe, qui permet de simplifier efficacement le problème.

Cette nouvelle fonction de valeur permet d'obtenir une meilleure approximation de la fonction de valeur optimale en s . En effet, si $\|V - V^*\|_\infty \leq \epsilon$ alors $\|V_H - V^*\|_\infty \leq \gamma^{H-1} \epsilon$ (cf. (Bertsekas & Tsitsiklis, 1996), section 2.3). Notons ici que la présence d'une fonction de valeur pour valuer les feuilles de l'arbre de décision est particulièrement importante dans la mesure où γ est proche de 1.

5 ANALYSE DES RÉSULTATS ET DISCUSSION

Les tableaux 2 et 3 synthétisent les résultats expérimentaux des diverses approches développées.

	Exp.	Exp. KW	App.	Arb.		
				H=2	H=3	H=4
Coût	1.25	0.98	1.06	1.01	0.98	0.90
Temps	16	16	20	196	2648	47690

TAB. 2 – Coûts estimés sur 500 trajectoires et temps de calcul à l'exécution pour le problème réduit

La première colonne désigne la politique experte initialement choisie et la politique experte optimisée par la méthode de Kiefer-Wolfowitz. La deuxième colonne désigne la meilleure politique obtenue par itération de la politique approchée. La troisième colonne désigne les politiques obtenues par recherche arbo-

2. Cette simplification vaut uniquement lors de la simulation pour la génération de l'arbre, lors de l'évolution "réelle" de la constellation, les pannes satellites surviennent suivant les probabilités indiquées en début de section.

	Exp. Init.	Exp. KW	App.	Arb.	
				H=2	H=3
Coût	1.72	1.29	1.86	1.86	-
Temps	23	23	299	32581	-

TAB. 3 – Coûts estimés sur 500 trajectoires et temps de calcul à l’exécution pour le problème complet

rescente sur un horizon H , les feuilles de l’arbre de décision étant valuées par la fonction de valeur obtenue par l’algorithme précédent.

5.1 Apprentissage purement critique

Pour les deux instances du problème, la politique dérivée de la fonction apprise en suivant la politique experte optimisée est satisfaisante bien que moins performante que la politique experte optimisée (surtout pour le problème complet). Les itérations suivantes de l’algorithme consistent à tenter d’améliorer cette politique experte de départ. Pour le problème réduit, on observe la convergence de l’itération de politique approchée vers une politique de qualité quasiment identique à celle de la politique experte optimisée. En revanche, pour le problème complet ces itérations n’ont pas permis d’améliorer significativement les résultats. Pire, les politiques de maintenance finissent par se dégrader en une politique “passive” qui ne prépare jamais de nouveaux lanceurs et laisse la constellation se dégrader.

Une explication possible à la relative inefficacité de l’itération de la politique approchée pour le problème complet réside vraisemblablement dans la mauvaise qualité de l’approximation de la fonction de valeur : l’imprécision de notre fonction de valeur ne permet pas à l’algorithme de prendre les décisions adéquates pour la préparation de nouveaux satellites et l’extension du plan de lancement, ainsi que pour le choix du plan orbital en cas de lancement.

5.2 Apprentissage purement acteur

L’algorithme de *Kiefer-Wolfowitz* est stoppé après un nombre d’itérations spécifié. La recherche d’un vecteur de paramètres optimal requiert quelques heures de calcul. Nous avons noté, que pour un nombre de simulations fixé, un compromis doit être trouvé entre le nombre de simulations effectuées à chaque itération afin d’estimer la fonction J et le nombre total d’itérations. Ainsi, à nombre de simulations constant, si nous effectuons plus de simulations à chaque itération, les estimations de la fonction J sont plus précises et la qualité de chaque mise à jour du vecteur paramètre θ meilleure, mais le nombre de ces mises à jour est réduit.

Nous avons obtenu une amélioration de la politique initiale (paramétrée par des valeurs sensées mais a priori non optimales), traduite par une diminution du coût moyen d’environ 20 % pour le problème réduit et de 23 % pour le problème complet.

5.3 Recherche arborescente

Pour le problème réduit, les résultats font apparaître une amélioration sensible à partir de $H=4$ mois, la qualité de la politique produite surpassant celle obtenue par optimisation d'une politique experte. Pour le problème complet, il est impossible d'évaluer la moyenne de la politique produite, en raison des temps de calculs induits pour $H > 2$ mois. Une évaluation très grossière pour $H = 3$ mois sur quelques trajectoires semble toutefois indiquer une politique similaire à la politique obtenue pour $H = 1$ mois et $H = 2$ mois c'est-à-dire que l'effort consenti ne semble pas rapporter de bénéfices. Toutefois, il semblerait cohérent que, comme pour le problème réduit, en considérant un horizon plus long - au prix de plusieurs semaines de calcul - la recherche arborescente permette d'améliorer les politiques produites.

5.4 Conclusions

Pour le problème complet, l'apprentissage purement acteur produit des résultats supérieurs à l'apprentissage purement critique et par un effort calculatoire moindre. Ceci tient sans doute au fait que nous avons été capables de définir pour ce problème une politique experte paramétrée efficace plus facilement qu'une représentation compacte efficace support d'une fonction de valeur.

La recherche arborescente n'a pas permis d'améliorer les résultats obtenus pour le problème complet, l'horizon sur lequel elle s'effectue étant probablement trop court. En revanche, pour le problème réduit, cette technique permet d'effectuer un saut qualitatif important en partant d'une fonction de valeur approximative. Elle permettrait de plus dans le cadre du problème réel de la constellation de satellites de tirer parti du temps on-line disponible (c'est-à-dire un mois, qui est le laps de temps entre deux décisions) pour développer l'arbre de décision sur un horizon plus long que celui envisagé jusqu'à présent dans nos tests. Une amélioration actuellement à l'étude consiste à ne développer l'arbre que partiellement en privilégiant les meilleures branches.

Une question centrale de l'apprentissage par renforcement est : vaut-il mieux effectuer une recherche directe dans l'espace des politiques ou apprendre une fonction de valeur définie sur l'espace d'états et de laquelle est déduite une politique ? Notre exemple illustre le fait que selon les caractéristiques du problème, il peut être avantageux de délaisser la fonction de valeur en utilisant une méthode faisant abstraction de la structure markovienne du problème ou au contraire qu'une meilleure - mais plus coûteuse - exploitation de la fonction de valeur est possible. Il souligne le fait que l'apprentissage par renforcement peut être combiné à des techniques de recherche en ligne, qui exploitent par simulation l'information que constitue l'état courant du système. La connaissance accumulée dans une fonction de valeur, même imparfaite, peut ainsi être mise à profit par l'exploration d'un arbre de décision.

RÉFÉRENCES

- ANDERSON C. W. (1999). *Approximating a Policy Can be Easier Than Approximating a Value Function*. Rapport interne, Computer Science Department, Colorado State University.
- AZAVIDAR F. (1999). Simulation optimization methodologies. In *Proc. of the 1999 Winter Simulation Conference, USA*.
- BARTO A. G., SUTTON R. S., & ANDERSON C. W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, **13**(5), 835–846.
- BAXTER J. & BARTLETT P. (1999). *Direct Gradient-Based Reinforcement Learning*. Rapport interne, Research School of Information Sciences and Engineering, Australian National University.
- BELLMAN R. E. (1957). *Dynamic Programming*. Princeton University Press.
- BERTSEKAS D. P. & TSITSIKLIS J. N. (1996). *Neuro-Dynamic Programming*. Belmont (MA): Athena Scientific.
- DAVIES S., NG A. Y. & MOORE A. (1998). Applying online search techniques to continuous-state reinforcement learning. In *AAAI/IAAI*, p. 753–760.
- GARCIA F., PÉRET L., TOMASINI L. & G.VERFAILLIE (2001). A Markov Decision Problem Approach for the Deployment and the Maintenance of a Satellite Constellation. In *Proceedings of the 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space (ISAIRAS 2001)*, Montréal.
- KEARNS M. J., MANSOUR Y. & NG A. Y. (1999). A sparse sampling algorithm for near-optimal planning in large markov decision processes. In *IJCAI*, p. 1324–1331.
- KIEFER J. & WOLFOVITZ J. (1952). Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics*, **23**, 463–446.
- MARBACH P. & TSITSIKLIS J. (1998). *Simulation-based optimization of markov reward processes*. Rapport interne, Lab. for Info. and Decision System, Massachusetts Institute of Technology, MA.
- PUTERMAN M. L. (1994). *Markov decision processes: discrete stochastic dynamic programming*. New York: Wiley-Interscience.
- ROBBINS H. & MONRO S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, **22**, 400–407.
- STRENS M. & MOORE A. W. (2001). Direct policy search using paired statistical tests. In *International Conference on Machine Learning*: Morgan Kaufmann.
- SUTTON R. S. & BARTO A. G. (1998). *Reinforcement Learning: An Introduction*. Cambridge, Massachusetts: MIT Press.
- SUTTON R. S., D.MCALLESTER, SINGH S. & MANSOUR Y. (1999). Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12, p. 1057–1063: MIT Press.
- WILLIAMS R. J. (1992). Simple statistical gradient-following algorithms for connectivist reinforcement learning. *Machine Learning*, **8**, 229–256.